CORTEX USERS GROUP

93, Long Knowle Lane, Wednesfield, Wolverhampton, West Midlands, WV11 1JG

Tel No: T Gray 0902 729078, E. Serwa 0902 732659

16

CORTEX USER GROUP NEWSLETTER (JAN 1988)

Issue Number 16

CONTENTS

REMEMBER TO SEND IN YOUR ARTICLES FOR THE NEXT NEWSLETTER

LETTERS

A.R.C. Badcock Hants

Users may like to know that the data separator I.C. the FDC9216B is available from M.S.Components Ltd on 01-670-4466 for £7.15 + VAT. I am interested in C.J. Youngs assembler is it significantly better to use than than R.M. Lees. Also is it possible to get a single density version of the MDEX boot track BOOT5D or BOOT5S.

C.J. Youngs assembler is written in machine code so it tends to run faster than R.M. Lees, also it will handle very large source files and programmes. It does have the disadvantage however of not producing a list file but this facility is to be added in the future. The MDEX system automatically selects single or double density formats by trying to read a disk and if it gets an error trying again in the other density mode. It should therefor be possible to use the boot track files in either density.

W.R. Bucknall Sheffield

I enclose copy of data manuals for Cannon MD110-220 drives for you to hold for other members that may require more information. Also Im looking forward to recieving MDEX Pascal from you as we are studying it here.

If any one would like a copy of the Cannon drives data manual please send £1.00 to cover photocopying.

W.D. Eaves Caithness

I have no mains switch for my Cortex Mk II and cannot find one that will fit the cutout, can you supply a suitable switch ?
What type of connector is required for the E.Bus. I have my Centronics interface connected by vero pins through the main P.C.B. in place of the connector. However I would like to add a backplane to enable the use of the Centronics board and other expansion boards but there is no way to get the E.Bus connector out of the case. Are you supposed to cut a hole ?
In some past newsletter there has been mentioned a communications package called Commtex is it still available and what hardware is required.

The mains switch for the Mk II Cortex is R.S. Components stock number 337-223 and is available from the user group for £1.00 inc The E.Bus connector is a DIN 41612 64 way a/c plug. The best way to connect up a backplane is to use a short length of ribbon cable from a mating socket. This will have to be passed through a hole cut in the side of the case. The Commtex package is now available from the group for £5.00 on disc. It will support all normal modems V21/23/24 etc and uses the serial port of the cortex for connection to the modem.

REMEMBER TO SEND IN YOUR ARTICLES FOR THE NEXT NEWSLETTER

CORTEX USERS GROUP NEWSLETTER 1988

Letters

Prem Holdaway

I have entered both of Mr Rudnicki's programmes, Missile command and Canyon, but have not been able to get any control keys to work even though I have checked my typing several times.

Has anyone else been able to get the listings of the two games to run ? If so please write in or even supply a copy of the programmes on disk.

O.C. Walden Milton Keynes

Congratulations on the aquisition of the MDEX software and enhanced coverage in the newsletter. I have been using the core system for some years now in particular for all my assembler programming. Perhaps now with affordable support we shall be able to eradicate some of the resident bugs. As you may know mdex files are not written to consecutive sectors on the disk but use interlaced sectors as defined in the Precsession table. I have written a routine to use this table to dump Mdex files in the correct sequence while in native Cortex mode. This obtains the full power of MDEX Editor ASM etc to write files and programmes for either mode. Would this be of interest to others.

I should think a lot of users would be interested please send more details. Also send in more details of any resident bugs you know of in MDEX so that we can try to find a solution.

P.J. Riddle Edinburgh

I am writing for the first time about my Cortex It has laid dorment for some time but seeing your newsletter has rekindled my interest. Mine is a much modified system but here are the basics.
Cortex with 9909 disk controller, 8" disk drives with a basic dos I wrote from scratch, Real time battery backed up clock, Battery backed up static ram, 20 meg hard disk interface (not yet completed), Eprom programmer (not yet completed).
As you can see I have been quite busy. And its a great pity the 99xx range has not prospered as it is a good standard. The reason for writing is ask if you can supply MDEX on 8" Bootable disks.

Yes indeed all MDEX and CDOS software can be supplied on 8" disks but only in single sided. I am sure many members would be interested in more details of your add-ons especially the hard disk interface and eprom programmer. Why not write an article or two for the newsletter.

Nigel Osmond Glostershire

Did MPE release the NOS operating system,which is the big brother of MDEX, to the User Group.

No up to date we do not have NOS for sale.

Letters

Dick Hall Scotland

I already have MDEX P.D.S. but do not have the source listing of MDEX.REL. I have modified my device drivers to accomodate a parralel printer but would also like to modify the main system programme if source is available.

As far as I know source code was never available for MDEX.REL but it should be possible to re-produce it using Anthony Rowell's dissasembler. If any other users have already done so could you please send a copy in to the user group.

D.L. Wright Fife

I have a Cortex running MDEX which I use to research into computer security. I am now planning to install a C.D.C. with a 36 Megabyte winchester and I am faced with drastic operating system changes. As I have 192K of memory available Stephen Pelc at M.P.E. has advised me to install NOS and use this to bring the winchester on line. I have lost touch with the User Group but now wish to re-join both to take advantage of the software availability and to make contributions to the Newsletter for this exellent machine. I also have a TMS 32010 evaluation module which uses a TMS 9995 to interface to the D.S.P.

At the moment we do not have NOS available from the User Group but if you can obtain it from M.P.E. I'm sure many of our readers would be interested in the results you get from interfacing a winchester drive. We look forward to hearing more from you in the future.

As you can see there has been quite a lot of interest in MDEX so far. I would like to take time to say thanks to Rex Collins who is trying to answer all queries that we recieve from MDEX users, and also to Anthony Rowell who has been of great assistance on the subject. The biggest burdon to us with MDEX distribution is photocopying the manuals. Prem Holdaway has volunteerd to type the MDEX user guide into a file that can be printed by the user. This would enable us to distribute manuals on disk. If anyone is interested in typing up any more manuals please let us know.

We have details of a few Cortex computers for sale on the second hand market, both Mk I and Mk II. Some have disk drives and other extras fitted and some software is included. Prices range from arround £50.00 to £180.00. If anyone would like to get hold of a second machine to use or just to keep for spares please let us know and we will pass on the information.

REMEMBER TO SEND IN YOUR ARTICLES FOR THE NEXT NEWSLETTER

PAINT and DUMP for Centronics 739 Printers

W.D. Eaves

Please find enclosed two listings which will be of use to people who use Centronics 739 printers. The first listing is a variation on the 'PAINT' program in Newsletter 4 and the second is a 'DUMP' program based upon the methods used in 'PAINT' The listings are the result of an exercise to convert the potentially useful programs into a form which I could use and also in order to learn more about programming in machine code.

PAINT

This program produces the same type of output as Tim Gray's 'PAINT' in Newsletter 4. An A4 size sheet is produced with each pixel mapped onto a 3x3 matrix depending upon its colour. The translation is exactly the same although the data in the table looks different. This is simply due to the way the data is prepared for the printer. The 3 bits representing pixel dots are stored in bits 1-3 of a byte in my program and bits 5-8 in the Epsom printer version. As with the original 'PAINT' the code is entirely relocatable.

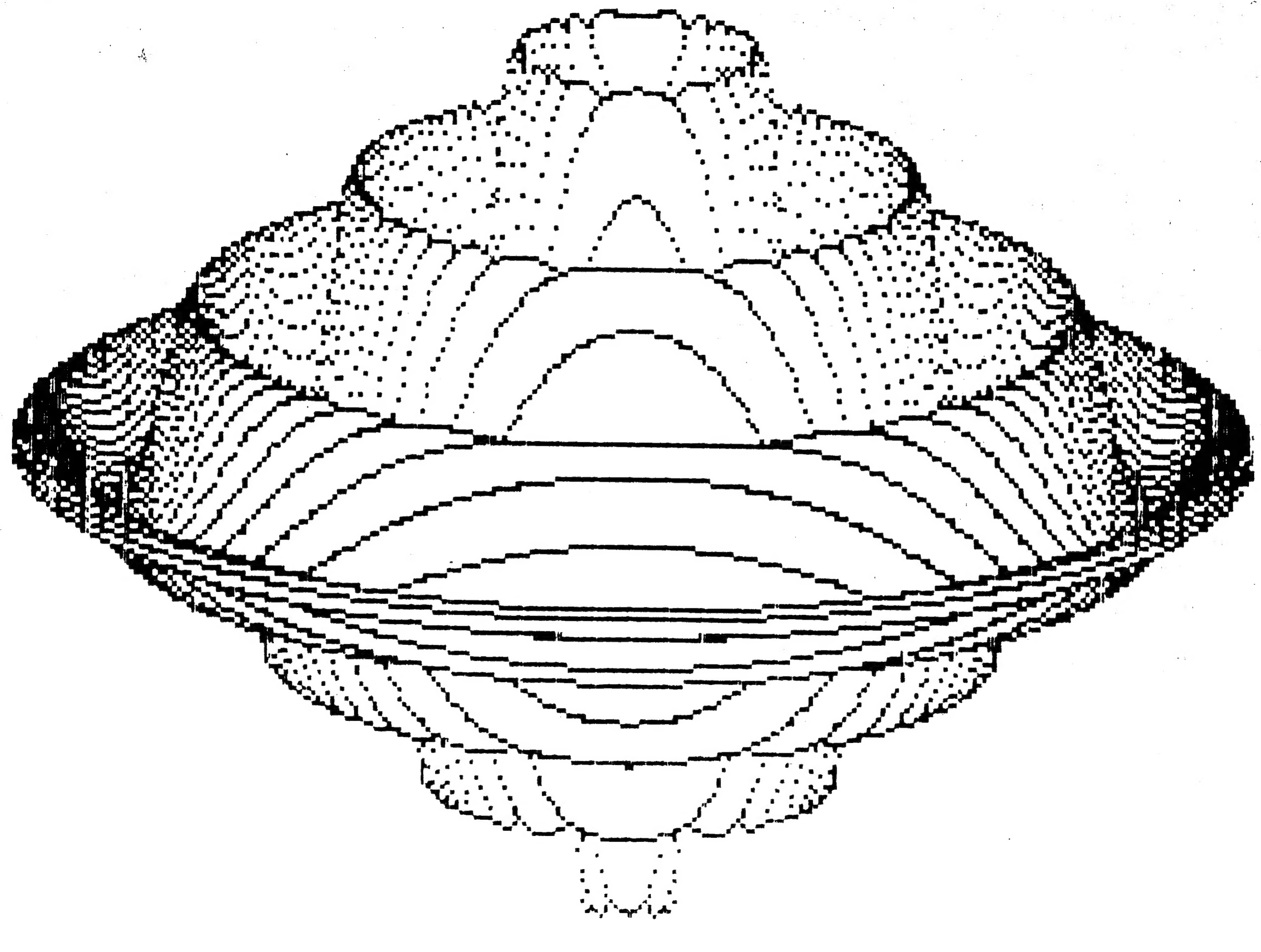
PAINT

```
5E00 020A LI    R10,>045B         }
5E04 068A BL    R10               }
5E06 C28B MOV   R11,R10           }
5E08 022B AI    R11,>004A         }   SET UP CODE TO BE RELOCATABLE
5E0C CA8B MOV   R11,@>00B2(R10)   }
5E10 CA8B MOV   R11,@>00E0(R10)   }
5E14 101F JMP   >5E54
5E16 0007 DATA >0007              }
5E18 0707 SETO R7                 }
5E1A 0707 SETO R7                 }
5E1C 0705 SETO R5                 }
5E1E 0205 LI    R5,>0500          }
5E22 0507 NEG   R7                }
5E24 0507 NEG   R7                }
5E26 0502 NEG   R2                }
5E28 0507 NEG   R7                }
5E2A 0207 LI    R7,>0007          }   COLOUR TABLE DATA.
5E2E 0005 DATA >0005              }
5E30 0205 LI    R5,>0500          }
5E34 0500 NEG   R0                }
5E36 0700 SETO R0                 }
5E38 0002 DATA >0002              }
5E3A 0007 DATA >0007              }
5E3C 0207 LI    R7,>0502          }
5E40 0500 NEG   R0                }
5E42 0200 LI    R0,>0000          }
5E46 0000 DATA >0000  }
5E48 0000 DATA >0000  } OLD UNIT FLAG
5E4A 0000 DATA >0000  }     X                     STORAGE FOR VARIABLES
5E4C 0000 DATA >0000  }     Y
5E4E 0000 DATA >0000  } OLD CURSOR POSITION
5E50 0000 DATA >0000  } COLOUR RETURNED FROM 'COL'.
5E52 0000 DATA >0000  }
5E54 0201 LI    R1,>00BF          }   SET Y=191
5E58 CA81 MOV   R1,@>0046(R10)    }
5E5C 1000 NOP
```

```
5E5E C060 MOV   @>0026,R1
5E62 1602 JNE   >5E68                    CHECK IF IN GRAPH MODE
5E64 2FA0 XOP   @>0030,14
5E68 CAA0 MOV   @>001E,@>0042(R10)       STORE OLD UNIT FLAG
5E6E 0201 LI    R1,>0008
5E72 C801 MOV   R1,@>001E                SET TO UNIT 4 : UNIT-1.
5E76 0201 LI    R1,>0A00
5E7A 0F01 WRIT  R1                       LINE FEED
5E7C 0201 LI    R1,>0D00
5E80 0F01 WRIT  R1
5E82 0201 LI    R1,>1B00
5E86 0F01 WRIT  R1                       ENABLE GRAPHICS ROUTINES.
5E88 0201 LI    R1,>2500
5E8C 0F01 WRIT  R1
5E8E 0201 LI    R1,>3000
5E92 0F01 WRIT  R1
5E94 1000 NOP
5E96 04EA CLR   @>0044(R10)              SET X=0
5E9A CAA0 MOV   @>EE36,@>0048(R10)       STORE OLD CURSOR POSITION
5EA0 1000 NOP
5EA2 04EA CLR   @>004A(R10)              CLEAR DATA STORE FOR COLOUR
5EA6 1000 NOP
5EA8 D82A MOVB  @>0045(R10),@>EE36       SET CURSOR TO X, Y
5EAE D82A MOVB  @>0047(R10),@>EE37
5EB4 0420 BLWP  @>1C9E                   GET COLOUR
5EB8 0000 DATA  >0000
5EBA D0AA MOVB  @>004A(R10),R2           STORE IN R2
5EBE 1604 JNE   >5EC8                    IF R2 = ∅
5EC0 D0A0 MOVB  @>0548,R2                THEN GET CURRENT BACKGROUND COLO
5EC4 0242 ANDI  R2,>0F00
5EC8 0982 SRL   R2,8                     PUT DATA IN LSB OF R2
5ECA 0203 LI    R3,>0003
5ECE 38C2 MPY   R2,R3                    GET APPROPRIATE PIXEL DATA.
5ED0 0224 AI    R4,>0011
5ED4 A10A A     R10,R4
5ED6 D174 MOVB  *R4+,R5
5ED8 D1B4 MOVB  *R4+,R6                  PIXEL DATA TO MSB OF R5, R6, R
5EDA D1F4 MOVB  *R4+,R7
5EDC B820 AB    @>1D49,@>EE36            CURSOR = X+1, Y
5EE2 0420 BLWP  @>1C9E
5EE6 0000 DATA  >0000
5EE8 D0AA MOVB  @>004A(R10),R2
5EEC 1604 JNE   >5EF6                    SAME AS 5EB4 TO 5EC8
5EEE D0A0 MOVB  @>0548,R2
5EF2 0242 ANDI  R2,>0F00
5EF6 0982 SRL   R2,8
5EF8 0935 SRL   R5,3
5EFA 0936 SRL   R6,3                     PIXEL DATA TO TOP 3 BITS OF LSB
5EFC 0937 SRL   R7,3
5EFE 0203 LI    R3,>0003
5F02 38C2 MPY   R2,R3
5F04 0224 AI    R4,>0011
5F08 A10A A     R10,R4                   SAME AS 5ECA TO 5EDA
5F0A D174 MOVB  *R4+,R5
5F0C D1B4 MOVB  *R4+,R6
5F0E D1F4 MOVB  *R4+,R7
```

```
5F10 0A35 SLA  R5,3
5F12 0A36 SLA  R6,3
5F14 0A37 SLA  R7,3
5F16 0225 AI   R5,>2000                  REARRANGE DATA FOR PRINTING.
5F1A 0226 AI   R6,>2000
5F1E 0227 AI   R7,>2000
5F22 0F05 WRIT R5
5F24 0F06 WRIT R6                        DATA TO PRINTER
5F26 0F07 WRIT R7
5F28 062A DEC  @>0046(R10)               Y = Y -1
5F2C 0201 LI   R1,>FFFF                  IF Y <> -1 THEN LOOP.
5F30 806A C    @>0046(R10),R1
5F34 16B9 JNE  >5EA8
5F36 0207 LI   R7,>0D00
5F3A 0F07 WRIT R7                        PRINT LINE FEED & CR.
5F3C 0207 LI   R7,>0A00
5F40 0F07 WRIT R7
5F42 0201 LI   R1,>00BF                  SET Y=191
5F46 CA81 MOV  R1,@>0046(R10)
5F4A 05EA INCT @>0044(R10)               X = X+2
5F4E 0201 LI   R1,>00FF                  IF X <= 255 THEN LOOP
5F52 806A C    @>0044(R10),R1
5F56 12A5 JLE  >5EA2
5F58 0201 LI   R1,>1B00
5F5C 0F01 WRIT R1                        DISABLE GRAPHICS ON PRINTER
5F5E 0201 LI   R1,>1300
5F62 0F01 WRIT R1
5F64 C82A MOV  @>0042(R10),@>001E        RESET OLD UNIT FLAG AND CURSO
5F6A C82A MOV  @>0048(R10),@>EE36        TO BASIC.
5F70 0380 RTWP
```

EXAMPLE OF DUMP OUTPUT



DUMP

This program produces a screen dump to printer using foreground & background colours only. This is useful for dumping screens of only two colours though a full coloured screen can be dumped to printer though no representation of colour can be seen. Each pixel is mapped to a 2x2 matrix which is either black or white only. Using a smaller matrix has the advantage of speed where no colour representation is required. Also the printout does not require rotation through 90 degrees which a 3x3 representation does.

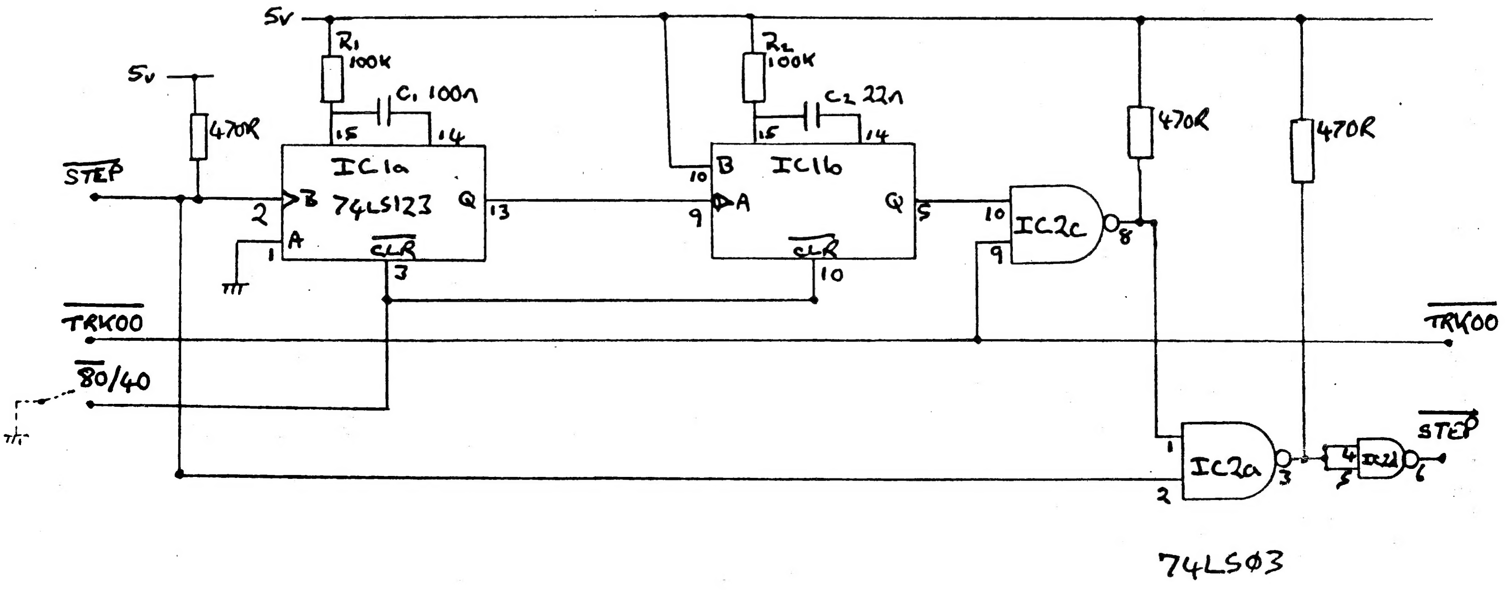
The notes on the listings should provide all further information as to how the routines work.

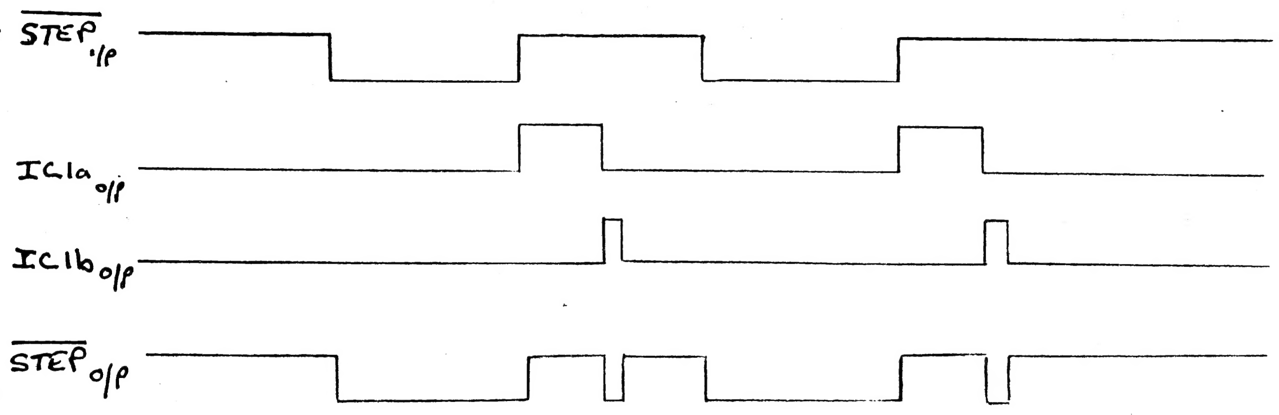
DUMP

```
5E00 020A LI   R10,>045B        
5E04 068A BL   R10              
5E06 C28B MOV  R11,R10                                  RELOCATABLE CODE
5E08 022B AI   R11,>0014        
5E0C CA8B MOV  R11,@>0116(R10)  
5E10 1007 JMP  >5E20
5E12 0000 DATA >0000         OLD UNIT
5E14 0000 DATA >0000            Y
5E16 0000 DATA >0000            X
5E18 0000 DATA >0000         OLD CURSOR                 VARIABLE STORAGE
5E1A 0000 DATA >0000         COLOUR FROM COL
5E1C 0000 DATA >0000         BACKGROUND COLOUR
5E1E 1000 NOP
5E20 04EA CLR  @>0010(R10)                              SET X = 0.
5E24 0201 LI   R1,>EE95                                 DISABLE COL CORRECTION
5E28 C801 MOV  R1,@>1D12
5E2C C060 MOV  @>0548,R1
5E30 0241 ANDI R1,>0F00                                 STORE CURRENT BACKGROUND COLOUR
5E34 CA81 MOV  R1,@>0016(R10)
5E38 C060 MOV  @>0026,R1
5E3C 1602 JNE  >5E42                                    CHECK IF IN GRAPH MODE
5E3E 2FA0 XOP  @>0030,14
5E42 CAA0 MOV  @>001E,@>000C(R10)
5E48 0201 LI   R1,>0008                                 SAVE OLD UNIT & SET NEW UNIT
5E4C C801 MOV  R1,@>001E
5E50 0201 LI   R1,>0A00
5E54 0F01 WRIT R1
5E56 0201 LI   R1,>0D00
5E5A 0F01 WRIT R1
5E5C 0201 LI   R1,>1B00                                 SET UP PRINTER
5E60 0F01 WRIT R1
5E62 0201 LI   R1,>2500
5E66 0F01 WRIT R1
5E68 0201 LI   R1,>3000
5E6C 0F01 WRIT R1
5E6E 1000 NOP
```

```
5E70 04EA CLR   @>000E(R10)                SET Y=0.
5E74 CAA0 MOV   @>EE36,@>0012(R10)         STORE OLD CURSOR
5E7A 1000 NOP
5E7C 04EA CLR   @>0014(R10)
5E80 D82A MOVB  @>000F(R10),@>EE37         CURSOR = X, Y
5E86 D82A MOVB  @>0011(R10),@>EE36
5E8C 1000 NOP
5E8E 06AA BL    @>0112(R10)                GET COLOUR -> R5
5E92 D142 MOVB  R2,R5
5E94 05A0 INC   @>EE36                     CURSOR = X, Y+1
5E98 0925 SRL   R5,2                       COLOUR -> BELOW MSB
5E9A 1000 NOP
5E9C 06AA BL    @>0112(R10)                COLOUR -> R5
5EA0 D142 MOVB  R2,R5
5EA2 05A0 INC   @>EE36                     CURSOR = X, Y+2
5EA6 0925 SRL   R5,2                       COLOUR -> BELOW MSB
5EA8 1000 NOP
5EAA 06AA BL    @>0112(R10)
5EAE D142 MOVB  R2,R5                      COLOUR -> R5
5EB0 ~~05A0 INC   @>EE36~~ NOP, NOP
5EB4 1000 NOP
5EB6 0A45 SLA   R5,4
5EB8 0225 AI    R5,>2000                   REARRANGE & PRINT DATA.
5EBC 0F05 WRIT  R5
5EBE 0F05 WRIT  R5
5EC0 05AA INC   @>0010(R10)
5EC4 0201 LI    R1,>00FF                   X = X + 1  IF X <= 255 THEN LOO
5EC8 806A C     @>0010(R10),R1
5ECC 12D7 JLE   >5E7C
5ECE 0207 LI    R7,>0A00
5ED2 0F07 WRIT  R7                         PRINT LF & CR
5ED4 0207 LI    R7,>0D00
5ED8 0F07 WRIT  R7
5EDA 1000 NOP
5EDC 04EA CLR   @>0010(R10)                SET X=0
5EE0 05AA INC   @>000E(R10)
5EE4 05EA INCT  @>000E(R10)                Y = Y+3  IF Y <= 191 THEN LOOP
5EE8 0201 LI    R1,>00BF
5EEC 806A C     @>000E(R10),R1
5EF0 12C7 JLE   >5E80
5EF2 0207 LI    R7,>1B00
5EF6 0F07 WRIT  R7                         RESET PRINTER
5EF8 0207 LI    R7,>1300
5EFC 0F07 WRIT  R7
5EFE C82A MOV   @>000C(R10),@>001E
5F04 C82A MOV   @>0012(R10),@>EE36         RESTORE UNIT CURSOR & COL
5F0A 0201 LI    R1,>F120                   CORRECTION
5F0E C801 MOV   R1,@>1D12                  TO BASIC
5F12 0380 RTWP
5F14 1000 NOP
5F16 1000 NOP
5F18 0420 BLWP  @>1C9E
5F1C 0000 DATA  >0000                      SUBROUTINE TO GET COLOUR.
5F1E C0EA MOV   @>0014(R10),R3
5F22 04C2 CLR   R2                         IF COL = BACKGROUND  R2 = 0
5F24 8A83 C     R3,@>0016(R10)                  ELSE            R2 = 3
5F28 1302 JEQ   >5F2E
5F2A 0202 LI    R2,>0300
5F2E 045B RT
5F30 1000 NOP
```

If your disk drives do not have double pulsing for 40/80 trk switching the circuit below can be used. As can be seen from the timing diagram the TMS 9909 outputs a squarewave of period 2*$\overline{\text{STEP}}$. The drive steps on the trailing (rising) edge, hence the second pulse must be generated a given time after the trailing edge when the $\overline{\text{STEP}}$ is inactive. The given time is stated in the manufacturers handbook as minimum time between $\overline{\text{STEP}}$ pulses. For Canon's MDD220 this is 3ms (210, & 110 6ms). Monostable, IC1a triggers on the trailing edge of the $\overline{\text{STEP}}$ pulse for a period of 0.45*R1*C1 (~4.5ms for the given components). The falling edge triggers the second monostable, IC1b which gives the second STEP pulse (~1ms for given components). The total period of the monostables must not be greater than $\overline{\text{STEP}}$ (as set up in CONFIG, 10ms in this case) as can be seen from the timing diagram. This second pulse is gated with $\overline{\text{TRK00}}$ to prevent head crashing (like head banging but without Marilion or Iron Maiden). It is then mixed in IC2a with the original $\overline{\text{STEP}}$ input to give the double pulse o/p. An open collector NAND is used to reduce chip count, hence the pull up resistors. Normal 80trk operation is achieved by holding the $\overline{\text{CLR}}$ i/p's low, preventing the monostables triggering.





More on Control Keys

W.R. Bucknall

Further to John's article on single keys for control in issue twelve page 13. All ASCII control characters, bar 5 are available as single keys (unaffected by CTRL or SHIFT without diodes) by connecting the new key-switches to the un-used matrix positions as listed below. Also listed are the functions of the other spare matrix positions (affected by CNTRL & SHIFT). Xn & Yn refer to the matrix positions as on the circuit diagram. As can be seen SO, SI, DC2, DC3, & DC4 are not available as single keys.

Matrix	ASCII		CONTROL	HEX	Used Positions
XØ,YØ	NUL	Null	^@	<ØØ>	
XØ,Y1	SOH	Start of Header	^A	<Ø1>	
XØ,Y2	STX	Start of Text	^B	<Ø2>	
XØ,Y3	ETX	End of Text	^C	<Ø3>	
XØ,Y4	EOT	End of Transmission	^D	<Ø4>	
	ENQ	Enquiry	^E	<Ø5>	EDIT key
XØ,Y6	ACK	Acknowledge	^F	<Ø6>	
XØ,Y7	BEL	Bell	^G	<Ø7>	
	BS	Backspace	^H	<Ø8>	← key
	HT	Horizontal Tab	^I	<Ø9>	→key
	LF	Line Feed	^J	<ØA>	↓ key
	VT	Vertical Tab	^K	<ØB>	↑ key
	FF	Form Feed	^L	<ØC>	CLEAR key
	CR	Carriage Return	^M	<ØD>	RETURN key
	SO	Shift Out	^N	<ØE>	no single key
	SI	Shift In	^O	<ØF>	no single key
X1,YØ	DLE	Data Link Escape	^P	<1Ø>	
XØ,Y8	DC1	Device Control 1	^Q	<11>	
	DC2	Device Control 2	^R	<12>	no single key
	DC3	Device Control 3	^S	<13>	no single key
	DC4	Device Control 4	^T	<14>	no single key
X1,Y5	NAK	Negative Acknoledge	^U	<15>	
	SYN	idle Synchronise	^V	<16>	INSERT key
	ETB	End of Tx'n Block	^W	<17>	DELETE key
X1,Y8	CAN	Cancel	^X	<18>	
X1,Y9	EM	End Medium	^Y	<19>	
X1,Y1Ø	SUB	Substitute	^Z	<1A>	
	ESC	Escape	^[	<1B>	ESCAPE key
X2,Y1	FS	Form Seperator	^\	<1C>	
X2,Y2	GS	Group Seperator	^]	<1D>	
	RS	Record Seperator	^^	<1E>	HOME key
X2,Y4	US	Unit Seperator	^_	<1F>	
	DEL	Delete		<7F>	RUBOUT key

Other unused matrix positions are as below, as normal,shift,control.

XØ,Y9	P	@	DLE
XØ,Y1Ø	O	_	SI
X1,Y1	K	[	VT
X1,Y2	L	\	FF
X1,Y3	N	^	SO
X1,Y4	M	]	CR
X2,Y5	<	<	NUL
X2,Y6	>	>	NUL
X2,Y7	,	,	NUL
X2,Y9	.	.	NUL
X3,Y3	_	DEL	US

MDEX software for the Cortex.

The article about MDEX software in the last newsletter has caused some mis-understanding. Firstly I did not intend to imply that the software was no longer copy-write protected. The whole of the system is still protected by copywrite of John Walker ex Marinchip Systems and Stephen Pelc of M.P.E. My note about the copy-write was just to set a price assuming any royalties payable were low enough for us to pay without having to adjust the price. In fact we have agreed to pay 20% of the selling price to M.P.E. for distribution to the apropriate writers.

The Forth and Nautilus cross compiler systems have not been released to the user group after all. Apparently they got mixed up the pile of disks collected from M.P.E. by mistake. They have now been withdrawn from our list of items for sale. We hope to have a public domain version of Fig-Forth available to run on CDOS format disks as an alterative in the near future.

MDEX software available is as follows :-

MDEX (Marinchip Disk Executive) is a disk operating system similar in some respects to CPM. It was originally developed by Marinchip in the U.S. for computers using the T.I. TMS9900 proccessor. It has been modified by M.P.E. in England for use on the Cortex.

MDEX CORE :- with Debug monitor, Text editor, Basic	£10.00
ASM & LINK :- Assembler and Linker	£10.00
SYSGEN :- System generation Kit	£10.00
WORD :- Word processor	£10.00
MDEX-PDS :- All of the above systems in one package	£30.00
SPL :- System programming language	£10.00
META :- Compiler generator	£10.00
QBASIC :- Basic compiler	£15.00
PASCAL :- Sequential Pascal	£10.00
WINDOW :- Full screen text editor	£15.00
SPELL :- Spelling checker	£10.00

All the above MDEX software is now available from the Cortex User Group at the normal address. All have good documentation, exept Pascal which has very little but many referances to published books are given.

Part 2.

QBASIC.

N D Osmond.

In this issue i will describe some of the other commands used in QBASIC,starting with the 'CHAIN' command. The CHAIN statement allows a QBASIC programme to pass controll to another programme,it may be another QBASIC programme or to one of the operating system utility programmes for example:- CHAIN "WINDOW"+TEXT.FILE following this statement the QBASIC programme would call the editor 'Window' and open the file called TEXT.FILE ready for editing.

String handling commands are plenty,with commands such as OVERLAY$ which will put part or all of one string into another.
Another is the "*" command:ie A$=" "*9 which will put nine blanks into the string A$.

As well as the single line function declaration Qbasic supports a Multiple line function,this means that after the function declaration ~~on the first~~ line any number of qbasic statements may make up the the function body.The function is ended by the FEND statement.Below is an extract from the qbasic manuel , the function takes two string arguments,LINE$ & WORD$,& returns an integer equal to the number of occurrences of WORD$ in LINE$.

```
 DEF FN.WORD.COUNT%(LINE$,WORD$)
      I%=0    {Occurrences found }
      K%=1    {Offset into string for search}
       WHILE 1
         J%=MATCH(UCASE$(WORD$)UCASE$(LINE$),K%)
       EXIT IF J%=0
         I%=I%+1
         K%=J%+LEN(WORD$)
       WEND
      FN.WORD.COUNT=I%
      FEND
```

The MATCH statement searches LINE$ for the pattern WORD$, UCASE$ converts all lower case characters in a string into upper case.

The above can be used as a subroutine or more important the function can be a subprogramme which would be compiled seperately, then linked to the main programme and called simply by the statement:-
DUMMEY%=FN.WORD.COUNT%,but more of this later,i would like to finish with the file input/output statements.

OPEN statement

```
      OPEN <expression> [RECL <expression> AS  <expression> [BUFF <exp>
```

The OPEN <exp>,exp=the file name
RECL <exp>,exp=record length,if used the file is random access
BUFF <exp>,exp=buffer size,if used the buffer space the file uses can be controlled,if a large buffer is allocated the performance of a programme can be improved by reducing the number of disc accesses,
AS <exp>,exp=file number 1 to 20 Also more then one file may be opened with one OPEN statement. eg:-

```
 OPEN "2/MYFILE" AS 1,"2/YOURFILE" AS 2
 OPEN "2/MYFILE" AS 1 BUFF 10
 OPEN "2/MYFILE" RECL 145 AS 1 ranom access
```

CREATE statement

```
       CREATE "2/MYFILE,130" AS 1 Will create a file 130*128 bytes
```

GETFILE statement

Will open the named file if it exists if it does not it will automatically create it safer to use than CREATE which would destroy a prevously created file.

READ statement;will read one or more varables from a sequential file
PRINT and PRINT USING statement;ill write to a file
IF END test end of file
GET & PUT similar to PRINT & READ but faster

A file may be opened and read then written to at any point in a sequential file or test for the end of a file then add to it. With the NOS operating system (MDEX's big brother-is it available ?) records can be locked.

The CALL & ADRS statement is used to envoke a assembly language module,values can be passed to the module from the main pgm and back again.

```
 ie:-CALL SRC(ADRS(VALUE%),COUNT%)
```

The assembly pgm SRC will be called & COUNT% & VALUE% will be passed.

VALUE% it's address will be passed to QBASIC's stack r10 this allows a value to be returned.

COUNT% the value in count% will be put on the stack.

Assembly language example from the QBASIC manual:-

```
          idt       "SRC"
          dstk      r10
.
src*      data      regg,src1         BLWP vector for entry
.
src1      mov       20(r13),r10       load caller's stack pointer
          popr      r0                pop value of COUNT% into r0
          popr      r1                pop address of VALUE% into r1
          mov       *r1,r2            load argument value
          src       r2                shift r2 by count in r0
          mov       r2,*r1            store back in VALUE% address
          mov       r10,20(r13)       udate QBASIC's stack pointer
          rtwp                        return to QBASIC
.
regg      bss       32                register workspace
.
          end
```

Next month linking with assembly pgms & linking QBASIC Modules.